# Requirements Processes That Work For Us

Kowalkowski, Paterno

# Why are we here?

In working with several experiments, we have a found a few techniques that seem to work well for recording and communicating requirements and project goals.

The purpose of this talk is to inform you of them.

# Types of “requirements”

*“Something wanted or needed. Something essential to the existence or occurrence of something else.”*
*– Merriam-Webster*

- We have *not* found it useful to draw sharp distinctions between “wants” and “needs”.
- Types of requirements include:
  - Constraints
  - Behavioral requirements
  - Speed requirements
  - Software engineering practices

# Successful techniques

- Use case narratives
- Road Map document
- Combination of the two

# Use case narratives

- What is a *use case*?
  - A formal description of someone or something interacting with a system explained as a series of steps
  - Purpose: capturing behavioral requirements
- What is a *use case narrative*?
  - An informal paragraph or two describing an interaction used to construct a use case
  - It is the first step in producing a formal “use case”
  - *It is valuable in and of itself*

# Advantages of UCN

- Approachable - right level of detail for:
  - Recording intended system behavior
  - Introductory documentation for new users
- Clearly documents user (or machine) interactions with the system
- Can easily capture acceptance criteria
  - Can form the basis for concrete tests

# Making use of UCN

- Users of any UCN collection will benefit from
  - A tool for sorting and viewing subsets or functionality
  - A dynamic repository to allow easy modification, addition, deletion, and reorganization
- Large sets are not necessarily good
  - Typical cases are important (illustrative examples)
  - Including “boundary conditions” or special cases
- Not everything is best described as a use case
  - expressing, as a use case, that gcc is required would be weird
- The collection should evolve as the system is better understood

# Road map

- What is it? A document containing a project's
  - Purpose
  - Requirements and use cases
  - Architecture
  - Technology choices (possible, rejected, accepted)
  - Record of decisions and associated arguments for and against them
  - Design elements
  - Performance and test criteria
  - Deployment plans
  - Administration plans
- A dynamic document – in support of an iterative process

# Advantages of road map

- History of decisions prevents needless rehashing of arguments
- Maintains focus on project completion
- Documentation of various aspects of a project are in close proximity
- Makes short-term goals apparent and priorities clear
- Helps in preparing a realistic schedule
- Provides an excellent introduction and guide for new members of the implementation team

# Making use of road map

- Use a document preparation system that enables concurrent development
- Requires discipline to keep it up to date
- Encourage joint ownership
- A template or outline can be easily constructed for new projects to follow
- Road maps for big projects can refer to ones for smaller projects

# Examples

- The RTES project used elements of both use case narratives and road map for their Demo2004 project
  - See BTeV document 3117
- The FroNtier project
- BTeV wiki

http://www-cdserver.fnal.gov/cd_public/cpd/aps/mfp/web/btev_sw/index.htm

# Issues arising when discussing requirements

- Coding guidelines and rules
- Design guidelines
- Documentation and good code examples
- Language choices

Our EDM/Framework notes available at

http://www-cdserver.fnal.gov/cd_public/cpd/aps/mfp/web/btev_sw/index.htm

may help in making decisions.

# Costs of decisions

- Example: Mixed language environment issues
  - Sharing of tools with other experiments may be more difficult
  - Infrastructure must be explicitly designed to handle this or it will not be possible to do it

**How is BTeV going to decide whether or not something is a requirement?**

**How is BTeV going to make, and enforce, the decision on a design that meets the requirement?**